Prof Simon McIntosh-Smith &

Dr Tom Deakin

HPC Research Group

University of Bristol

Early SYCL results from the Bristol Performance Portability Study

University of Bristol

http://uob-hpc.github.io http://hpc.tomdeakin.com

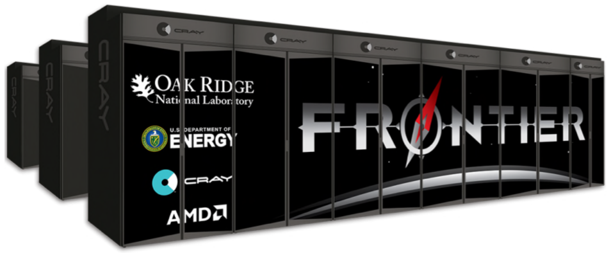
https://uob-hpc.github.io/SimonMS/

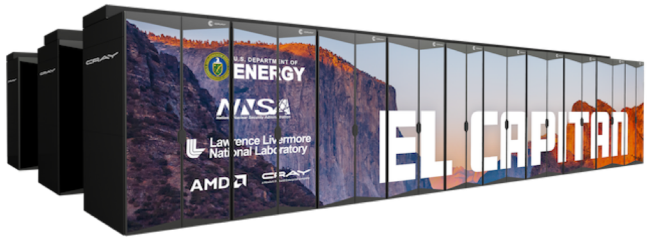
GW4

Challenges at Exascale

- The coming generation of Exascale supercomputers will contain a diverse range of architectures at massive scale

  - **Perlmutter**: AMD EYPC CPUs and NVIDIA GPUs (pre-Exascale)
  - **Frontier**: AMD EPYC CPUs and Radeon GPUs
  - **Aurora**: Intel Xeon CPUs and Xe GPUs
  - **El Capitan**: AMD EPYC CPUs and Radeon GPUs
  - **Fugaku**: Fujitsu A64fx Arm CPUs




U.S. DEPARTMENT OF ENERGY
NNSA
Lawrence Livermore National Laboratory
AMD
CRAY
EL CAPITAN

The Next Platform, Jan 13th 2020: "HPC in 2020: compute engine diversity gets real"
https://www.nextplatform.com/2020/01/13/hpc-in-2020-compute-engine-diversity-gets-real/

Recent architectural trends

- CPUs have evolved to include **lots of cores** and **wide vector units**
- 32 core CPUs been around for a while (AMD Naples, Marvell ThunderX2)
- 48, 64 core CPUs have now arrived (A64FX, Rome)
- Chiplet manufacturing processes likely to be an important future trend
- **Renewed competition in CPUs** is crucial to the health of the HPC ecosystem, and for performance per dollar

- GPUs incorporating latest memory technologies (HBM)
  - So does A64FX CPUs (and so did KNL)
- GPUs have **lots of cores** and very **wide vector units**
- Lightweight cores becoming more complex (caches, specialised accelerators, etc)
- Vendor competition increasing (AMD GPUs in Frontier and El Capitan, Intel GPUs in Aurora, NVIDIA GPUs in pre-Exascale Perlmutter)

University of BRISTOL

GW4

https://uob-hpc.github.io/publications/

The International Journal of High Performance Computing Applications
2015, Vol. 29(2) 119–134
© The Author(s) 2014
Reprints and permissions: sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/1094342014528252
hpc.sagepub.com
SAGE

Original Article

High performance in silico virtual drug screening on many-core processors

Simon McIntosh-Smith[1], James Price[1], Richard B Sessions[2] and Amaurys A Ibarra[2]

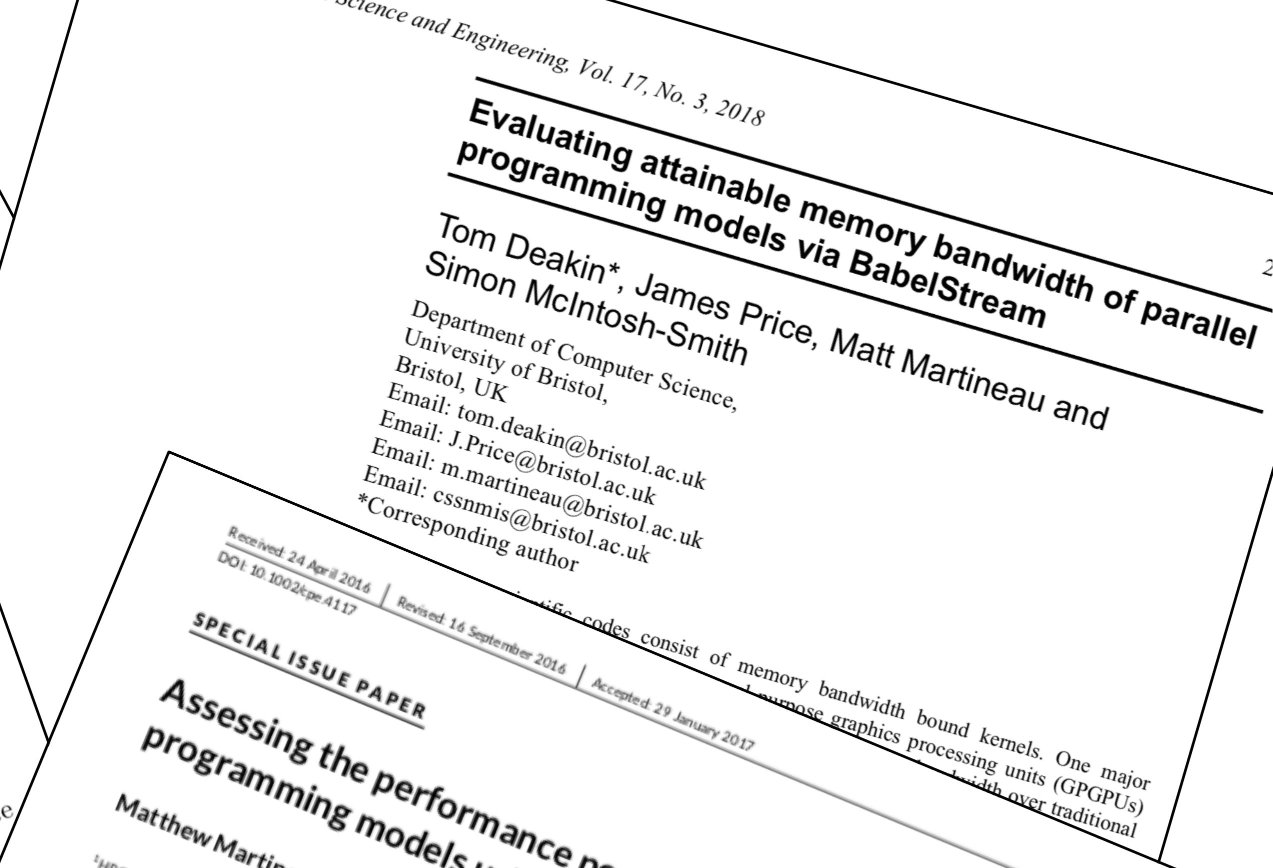
Abstract
Drug screening is an important part of the drug development pipeline for the pharmaceutical industry. Tradition
base … being augmented with computational methods, ranging from simple molecula
sear … matching to more computationally intensive approaches,
cula … to their targets, typically protein
wo … to the Op
da
op
pe
e
c

On the Performance Portability of Structured Grid Codes on Many-Core Computer Architectures

Simon McIntosh-Smith, Michael Boulton, Dan Curran, and James Price

Department of Computer Science, University of Bristol,
Woodland Road, Clifton, Bristol, BS8 1UB, UK
http://www.cs.bris.ac.uk/home/simonm/

Abstract. With the advent of many-core computer arc
as GPGPUs from NVIDIA and AMD, and more rece
Phi, ensuring performance portability of HPC c
becoming more complex. In this work we have f
tant application area — structured grid codes
iques for ensuring performance portability a
high-end many-core architectures. We c
lattice Boltzmann code (D3Q19 BG
lication from Sandia's Mantevo
quality structured grid, mu
have developed OpenC
functional port
ns of these st
ing hybri


Science and Engineering, Vol. 17, No. 3, 2018

Evaluating attainable memory bandwidth of parallel programming models via BabelStream

Tom Deakin*, James Price, Matt Martineau and Simon McIntosh-Smith
Department of Computer Science,
University of Bristol,
Bristol, UK
Email: tom.deakin@bristol.ac.uk
Email: J.Price@bristol.ac.uk
Email: m.martineau@bristol.ac.uk
Email: cssnmis@bristol.ac.uk
*Corresponding author

… scientific codes consist of memory bandwidth bound kernels. One major
… purpose graphics processing units (GPGPUs)
… bandwidth over traditional

Received: 24 April 2016 | Revised: 16 September 2016 | Accepted: 29 January 2017
DOI: 10.1002/cpe.4117

SPECIAL ISSUE PAPER

WILEY

Assessing the performance portability of modern parallel programming models using TeaLeaf

Matthew Martineau[1] | Simon McIntosh-Smith[1] | Wayne Gaudin[2]

[1]HPC Group, University of Bristol, Bristol, UK
[2]UK Atomic Weapons Establishment (AWE), Aldermaston, UK

Correspondence
Matthew Martineau, Merchant Venturers Building, Woodland Road, Bristol, BS8 1UB, UK
Email: m.martineau@bristol.ac.uk

Funding information
UK Atomic Weapons Establishment; Engineering and Physical Sciences Research Council (EPSRC)

Summary
In this work, we evaluate several emerging parallel programming models: Kokkos, RAJA, OpenACC, and OpenMP 4.0, against the mature CUDA and OpenCL APIs. Each model has been used to port Tealeaf, a miniature proxy application, or mini app, that solves the heat conduction equation and belongs to the Mantevo Project. We find that the best performance is achieved with architecture-specific implementations but that, in many cases, the performance portable models are able to solve the same problems to within a 5% to 30% performance penalty. While the models expose varying levels of complexity to the developer, they all achieve reasonable performance with this application. As such, if this small performance penalty is permissible for a domain, we believe that productivity and development complexity can be co… differentiators when choosing a modern parallel programming mod… Tealeaf

KEYWORDS
Kokkos, OpenMP 4.0, perform

University of Bristol

What do we mean by "performance portability?"

"A code is performance portable if it can achieve a similar fraction of peak hardware performance on a range of different target architectures."

Questions:

- **Does it have to be a "good" fraction?** YES! Ideally within 20% of "best achievable", i.e. of hand-optimized OpenMP, CUDA, …
- **How wide is the range of target architectures?** Depends on your goal, but important to allow for future architectural developments

A systematic evaluation of Performance Portability

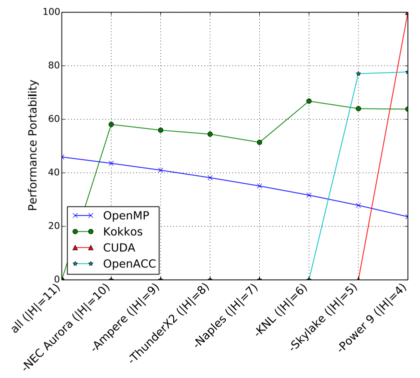
- Studying Performance Portability is ***hard***!
  - Must be **rigorous** about doing as well as possible across a wide range issues: architectures, programming languages, algorithms, compilers, …
- It takes a lot of effort to do this well
- Motivated by our results so far, in Bristol we have initiated a wide-ranging evaluation of Performance Portability:
  - Across many codes
  - Across many programming languages
  - Across many architectures
- Our goal is to share these codes and results to further the fundamental understanding of performance portability

https://doi.org/10.1109/P3HPC49587.2019.00006

TeaLeaf

Lower is better

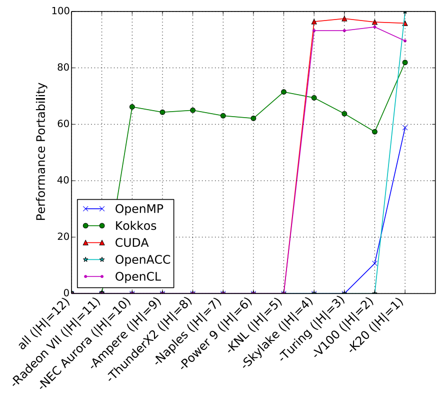
	OpenMP	Kokkos	CUDA	OpenACC
Skylake	317	370	-	-
KNL	191	885	-	-
Power 9	254	393	-	341
Naples	348	372	-	-
ThunderX2	314	439	-	-
Ampere	793	892	-	-
NEC Aurora	79.1	-	-	-
K20	1605	712	445	629
P100	190	187	122	153
V100	281	127	81.0	103
Turing	962	181	116	139



CloverLeaf

Lower is better

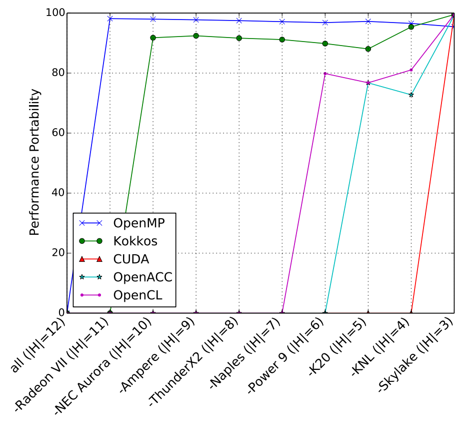
	OpenMP	Kokkos	CUDA	OpenACC	OpenCL
Skylake	376	463	-	877	-
KNL	250	666	-	698	-
Power 9	376	544	-	768	-
Naples	327	395	-	337	-
ThunderX2	457	772	-	-	-
Ampere	1309	1452	-	-	-
NEC Aurora	323	-	-	-	-
K20	9737	1297	592	-	572
P100	226	163	139	133	149
V100	-	108	88.8	90.1	97.9
Turing	-	211	213	199	213
Radeon VII	-	-	-	-	106



BabelStream

Higher is better

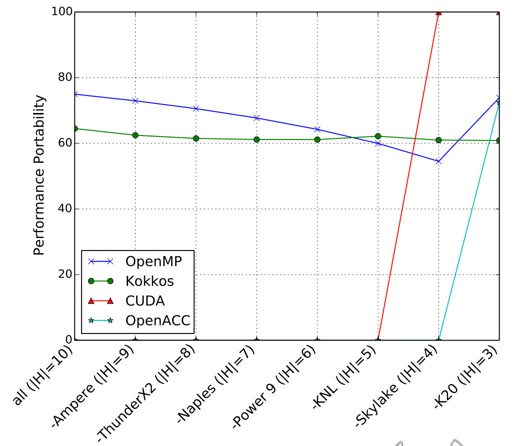
	OpenMP	Kokkos	CUDA	OpenACC	OpenCL
Skylake	80.2%	68.1%	-	32.4%	41.8%
KNL	92.2%	62.1%	-	90.7%	58.4%
Power 9	72.8%	73.6%	-	72.5%	-
Naples	65.9%	62.7%	-	-	-
ThunderX2	85.3%	84.7%	-	-	-
Ampere	66.4%	57.3%	-	-	-
NEC Aurora	81.3%	-	-	-	-
K20	69.2%	72.9%	72.3%	-	72.8%
P100	75.5%	76.1%	75.4%	75.3%	75.3%
V100	86.0%	92.0%	92.6%	92.1%	93.2%
Turing	85.7%	90.0%	90.2%	90.1%	89.9%
Radeon VII	-	-	-	-	79.4%



MiniFMM

Lower is better

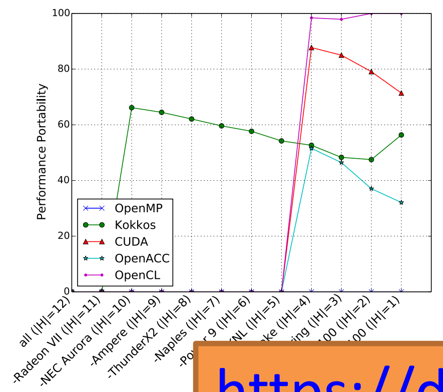
	OpenMP	Kokkos	CUDA	OpenACC
Skylake	8.7	12.9	-	-
KNL	11.4	20.2	-	-
Power 9	23.6	38.5	-	-
Naples	13.1	20.5	-	-
ThunderX2	21.9	30.6	-	-
Ampere	116	127	-	-
K20	56.7	28.2	17.3	-
P100	5.0	4.7	3.5	4.3
V100	3.1	4.4	2.5	3.8
Turing	3.2	4.2	2.3	3.2



Neutral

Lower is better

	OpenMP	Kokkos	CUDA	OpenACC	OpenCL
Skylake	8.0	13.0	-	-	-
KNL	23.8	28.1	-	-	-
Power 9	8.3	11.1	-	-	-
Naples	14.5	16.6	-	-	-
ThunderX2	12.6	13.5	-	-	-
Ampere	37.4	43.3	-	-	-
NEC Aurora	2784	-	-	-	-
K20	-	52.7	41.6	92.5	29.7
P100	-	9.5	4.4	8.9	3.9
V100	-	6.2	3.1	3.3	3.3
Turing	-	9.3	6.9	8.7	6.7
Radeon VII	-	-	-	-	3.7



University of BRISTOL

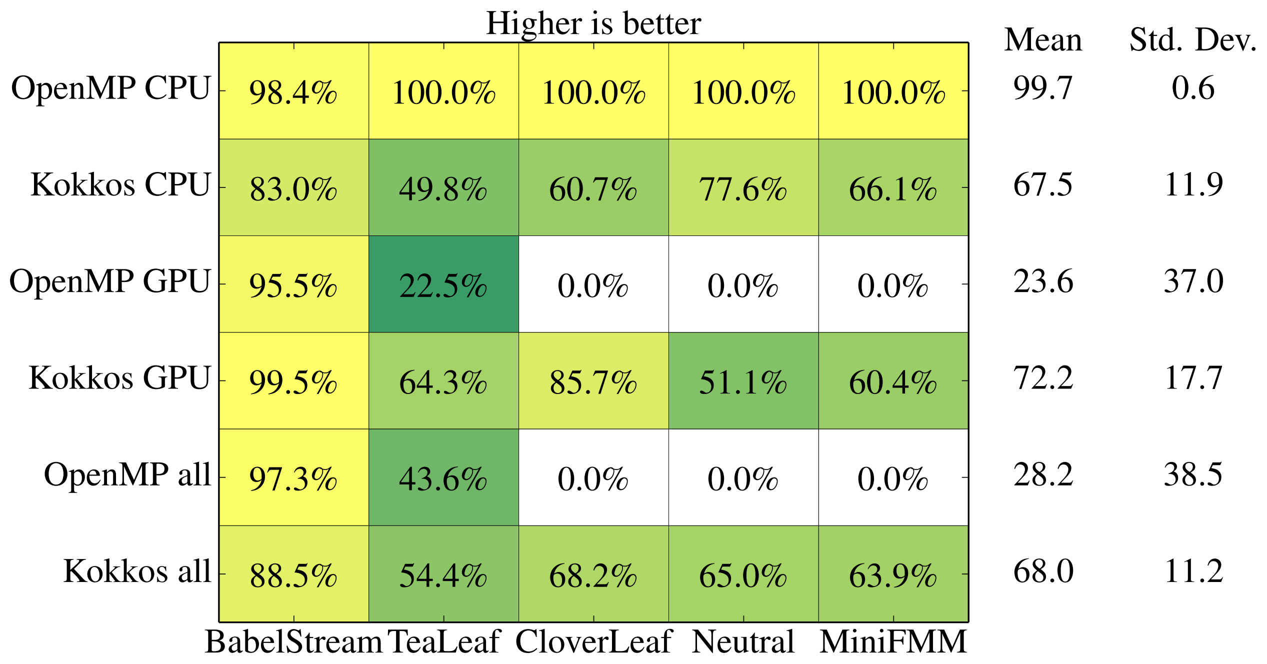
https://doi.org/10.1109/P3HPC49587.2019.00006

http://uob-hpc.github.io
http://hpc.tomdeakin.com

GW4

Performance Portability of OpenMP and Kokkos

- Heatmap shows PP metric on chosen platform subsets
- Rows indicate how a model fairs across different applications
- OpenMP achieving best performance on CPUs but struggles on GPUs due to support
- Kokkos shows a small overhead on CPUs
  - PP metric tells us to expect the abstraction of OpenMP/CUDA to reduce performance by ~15-50%



Higher is better

	BabelStream	TeaLeaf	CloverLeaf	Neutral	MiniFMM	Mean	Std. Dev.
OpenMP CPU	98.4%	100.0%	100.0%	100.0%	100.0%	99.7	0.6
Kokkos CPU	83.0%	49.8%	60.7%	77.6%	66.1%	67.5	11.9
OpenMP GPU	95.5%	22.5%	0.0%	0.0%	0.0%	23.6	37.0
Kokkos GPU	99.5%	64.3%	85.7%	51.1%	60.4%	72.2	17.7
OpenMP all	97.3%	43.6%	0.0%	0.0%	0.0%	28.2	38.5
Kokkos all	88.5%	54.4%	68.2%	65.0%	63.9%	68.0	11.2

- Final row here (Kokkos all) shows performance portability is possible
  - Mean and standard deviation shows we would expect Kokkos to achieve 59-79% of best application performance on average

https://doi.org/10.1109/P3HPC49587.2019.00006

University of BRISTOL

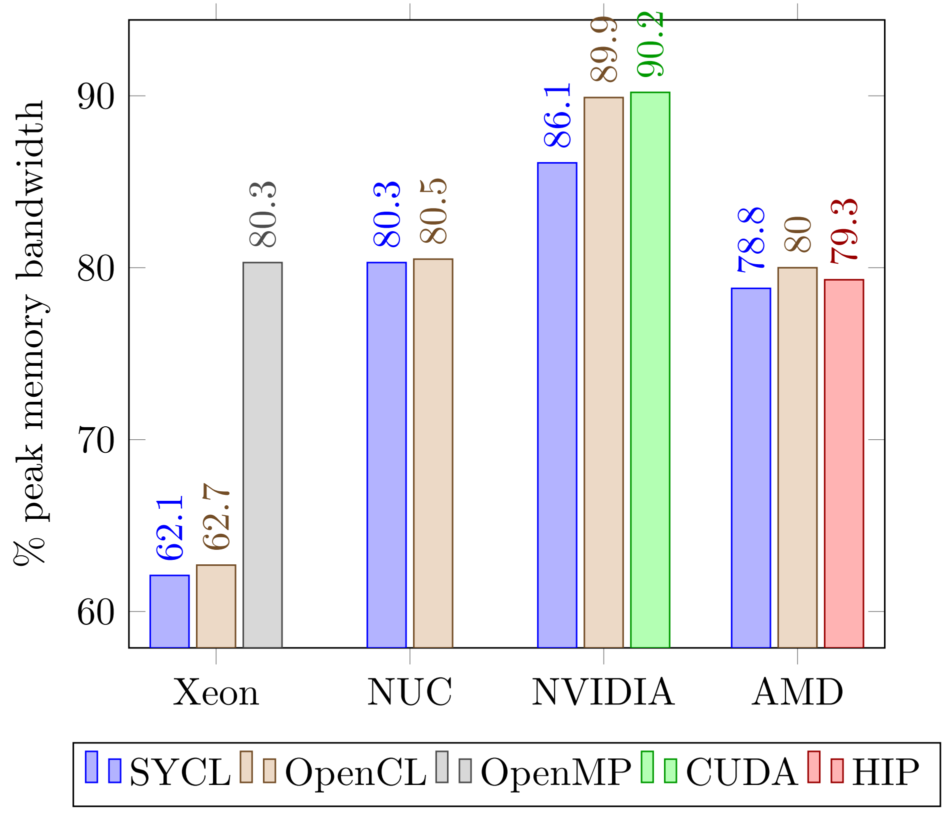
http://uob-hpc.github.io
http://hpc.tomdeakin.com

SYCL

- SYCL is a single-source C++ parallel programming model for heterogenous platforms from Khronos
  - Open standard
  - Modern C++
  - Commercial support from Intel with oneAPI/DPC++ and Codeplay
  - Open-source support growing to support wider set of platforms
- One possible option for programming CPUs, GPUs, etc. in a performance portable way

Performance Portability of SYCL

- Paper at IWOCL explored performance on Intel CPUs and GPUs from Intel, AMD and NVIDIA.
  - Comparisons with OpenCL, OpenMP, CUDA and HIP
  - Very promising results so far, but more work to do in the HPC ecosystem
  - Intel's OpenCL runtime on CPUs has known issues which hopefully will improve as part of oneAPI

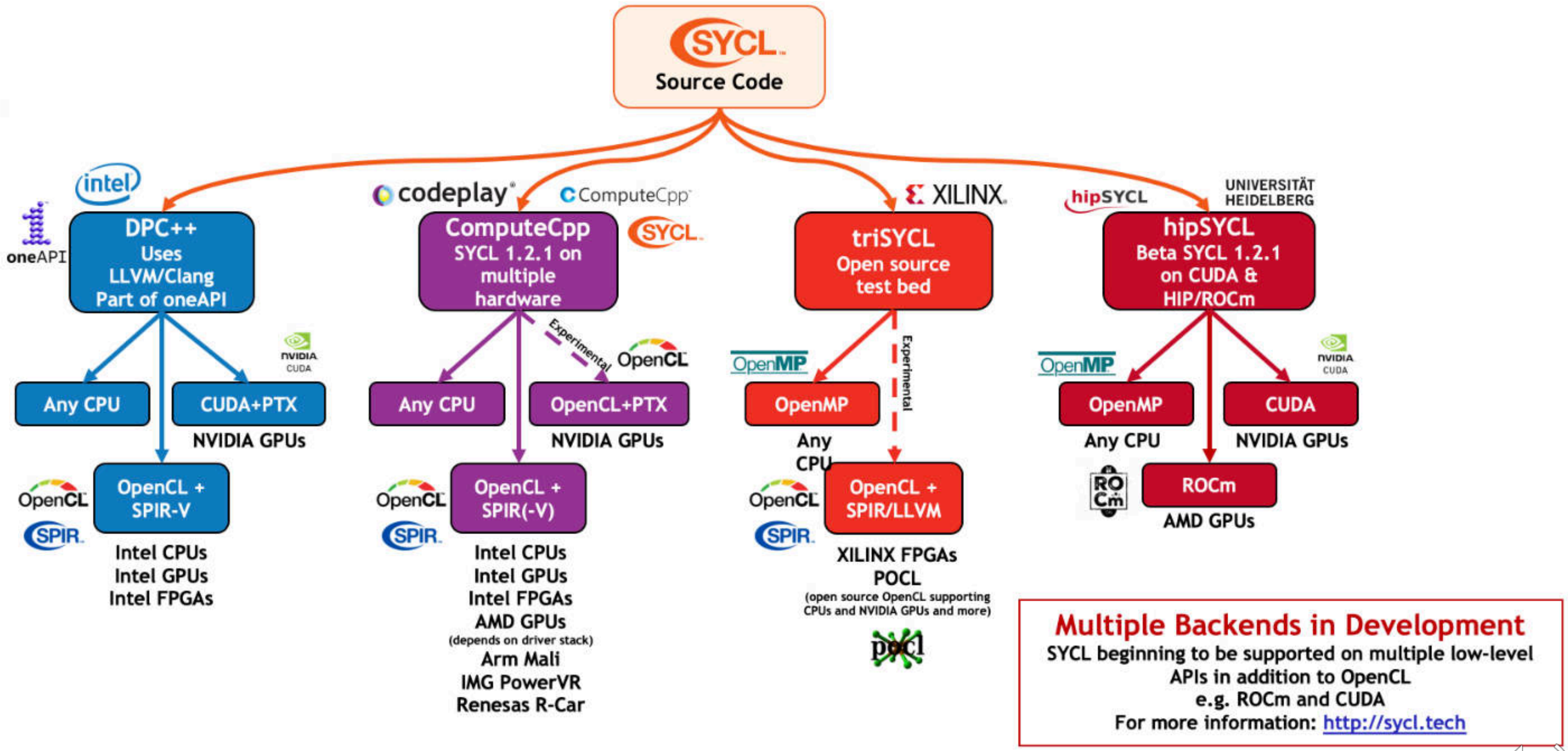


BabelStream Triad

https://doi.org/10.1145/3388333.3388643

http://uob-hpc.github.io
http://hpc.tomdeakin.com

Image from https://www.khronos.org/sycl/

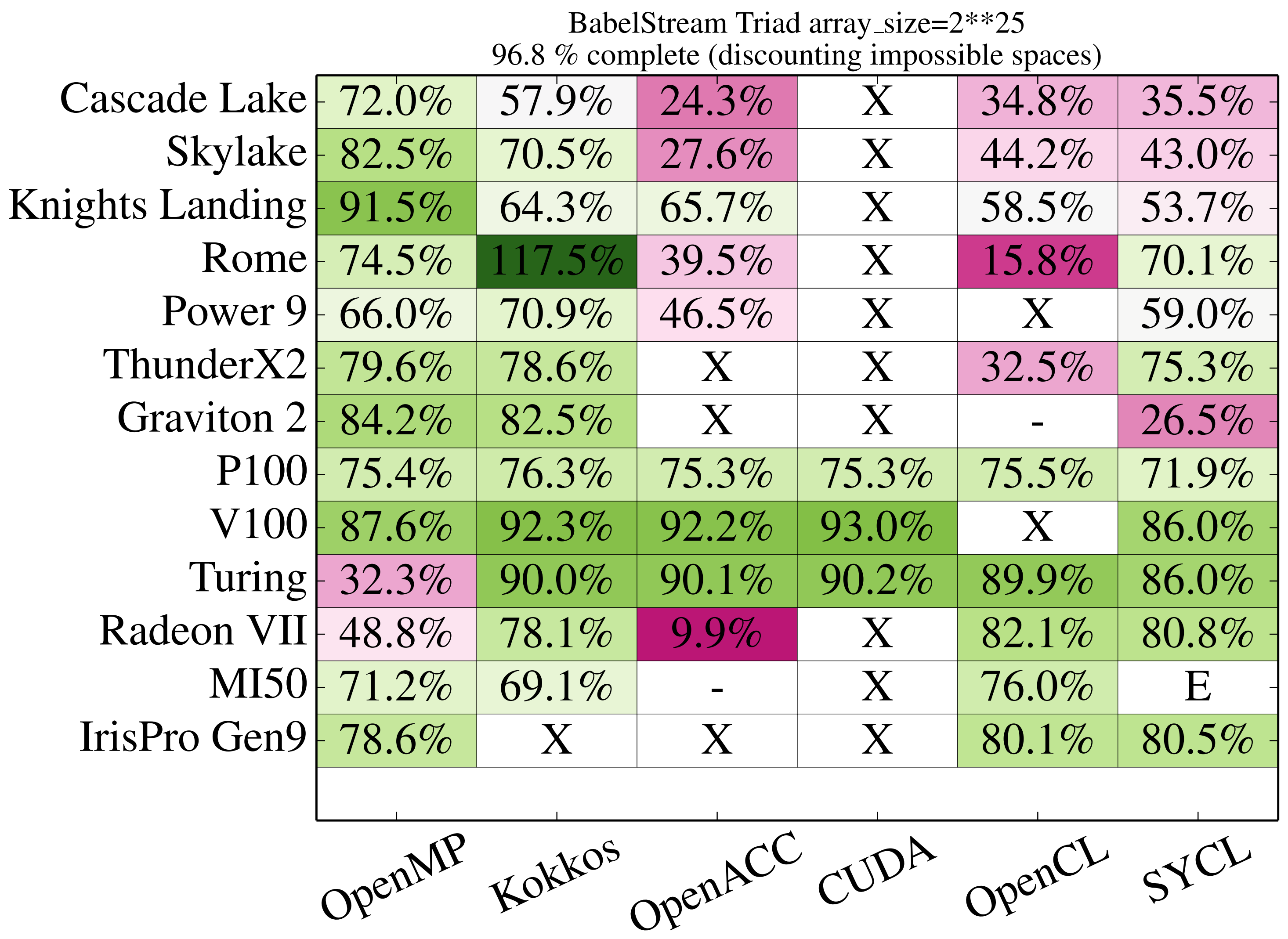


Multiple Backends in Development

SYCL beginning to be supported on multiple low-level APIs in addition to OpenCL e.g. ROCm and CUDA

For more information: http://sycl.tech

Latest BabelStream results



BabelStream Triad array_size=2**25
96.8 % complete (discounting impossible spaces)

	OpenMP	Kokkos	OpenACC	CUDA	OpenCL	SYCL
Cascade Lake	72.0%	57.9%	24.3%	X	34.8%	35.5%
Skylake	82.5%	70.5%	27.6%	X	44.2%	43.0%
Knights Landing	91.5%	64.3%	65.7%	X	58.5%	53.7%
Rome	74.5%	117.5%	39.5%	X	15.8%	70.1%
Power 9	66.0%	70.9%	46.5%	X	X	59.0%
ThunderX2	79.6%	78.6%	X	X	32.5%	75.3%
Graviton 2	84.2%	82.5%	X	X	-	26.5%
P100	75.4%	76.3%	75.3%	75.3%	75.5%	71.9%
V100	87.6%	92.3%	92.2%	93.0%	X	86.0%
Turing	32.3%	90.0%	90.1%	90.2%	89.9%	86.0%
Radeon VII	48.8%	78.1%	9.9%	X	82.1%	80.8%
MI50	71.2%	69.1%	-	X	76.0%	E
IrisPro Gen9	78.6%	X	X	X	80.1%	80.5%

- Showing architectural efficiency
  - Percentage of peak memory bandwidth
- Latest and greatest CPUs and GPUs from all vendors
- (Near) complete coverage for OpenMP, Kokkos and SYCL!
  - Much better coverage than our previous study

University of BRISTOL

http://uob-hpc.github.io
http://hpc.tomdeakin.com

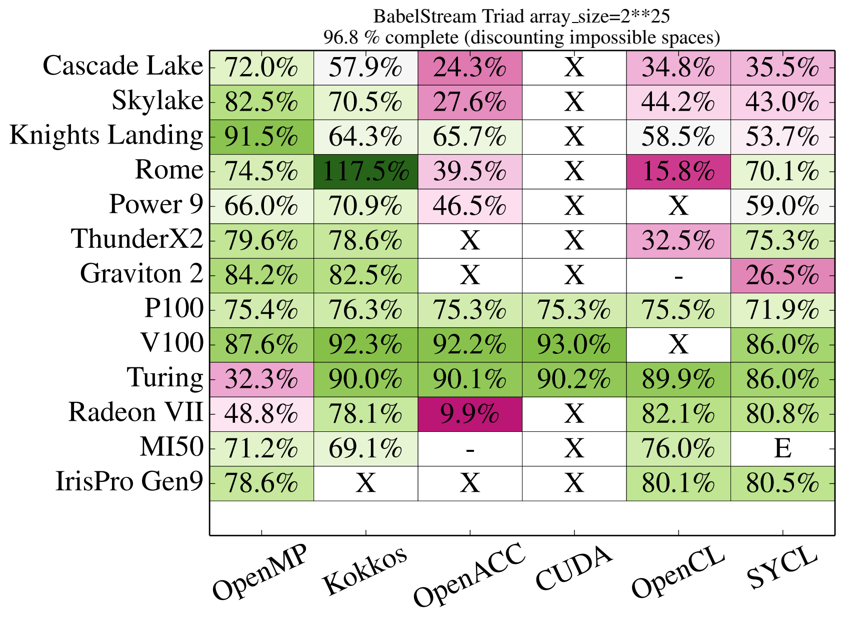
GW4

S.J. Pennycook, J.D. Sewall, V.W. Lee, *A metric for performance portability*, in Proceedings of the International Workshop on Performance Modelling, Benchmarking and Simulation, 2016. URL: http://arxiv.org/abs/1611.07409.

Performance portability of BabelStream

- Using Performance Portability metric (from Pennycook/Sewall/Lee), Kokkos and SYCL still score 0 due to single missing result.
- Two approaches to work around this (both have similar effect) :
  - Calculate metric excluding the missing result.
  - Remove unsupported platforms.

PP metric	OpenMP	Kokkos	SYCL
All platforms	67.4	0.0	0.0
Excluding missing data in each model	67.4	76.5	56.0
Excluding MI50 and Iris Pro 580 for all models	66.2	77.3	54.5

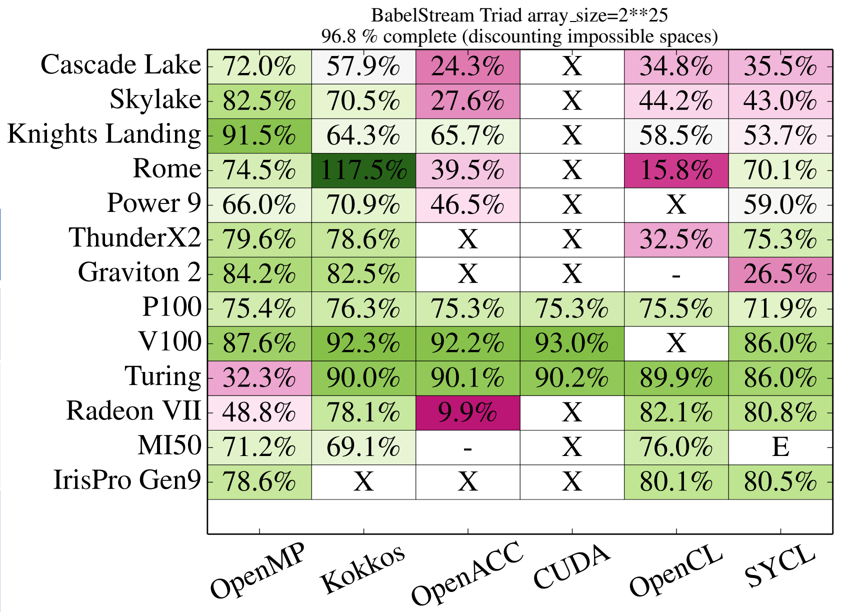

BabelStream Triad array_size=2**25
96.8 % complete (discounting impossible spaces)

	OpenMP	Kokkos	OpenACC	CUDA	OpenCL	SYCL
Cascade Lake	72.0%	57.9%	24.3%	X	34.8%	35.5%
Skylake	82.5%	70.5%	27.6%	X	44.2%	43.0%
Knights Landing	91.5%	64.3%	65.7%	X	58.5%	53.7%
Rome	74.5%	117.5%	39.5%	X	15.8%	70.1%
Power 9	66.0%	70.9%	46.5%	X	X	59.0%
ThunderX2	79.6%	78.6%	X	X	32.5%	75.3%
Graviton 2	84.2%	82.5%	X	X	-	26.5%
P100	75.4%	76.3%	75.3%	75.3%	75.5%	71.9%
V100	87.6%	92.3%	92.2%	93.0%	X	86.0%
Turing	32.3%	90.0%	90.1%	90.2%	89.9%	86.0%
Radeon VII	48.8%	78.1%	9.9%	X	82.1%	80.8%
MI50	71.2%	69.1%	-	X	76.0%	E
IrisPro Gen9	78.6%	X	X	X	80.1%	80.5%

Performance portability of BabelStream on CPUs and GPUs

- Compute the metric for each model (where we have results) on CPUs only and GPUs only.
- Kokkos still strong on both classes of device.
- OpenMP GPU support better but still room for improvement.
- SYCL support on CPUs needs improvement to resolve:
  - NUMA and thread placement issues of OpenCL backends.
  - Parallelism mapping of OpenMP backends.

PP metric	OpenMP	Kokkos	SYCL
Excluding missing data in each model	67.4	76.5	56.0
Supported CPUs only	77.8	74.1	46.0
Supported GPUs only	58.3	80.2	80.7


BabelStream Triad array_size=2**25
96.8 % complete (discounting impossible spaces)

	OpenMP	Kokkos	OpenACC	CUDA	OpenCL	SYCL
Cascade Lake	72.0%	57.9%	24.3%	X	34.8%	35.5%
Skylake	82.5%	70.5%	27.6%	X	44.2%	43.0%
Knights Landing	91.5%	64.3%	65.7%	X	58.5%	53.7%
Rome	74.5%	117.5%	39.5%	X	15.8%	70.1%
Power 9	66.0%	70.9%	46.5%	X	X	59.0%
ThunderX2	79.6%	78.6%	X	X	32.5%	75.3%
Graviton 2	84.2%	82.5%	X	X	-	26.5%
P100	75.4%	76.3%	75.3%	75.3%	75.5%	71.9%
V100	87.6%	92.3%	92.2%	93.0%	X	86.0%
Turing	32.3%	90.0%	90.1%	90.2%	89.9%	86.0%
Radeon VII	48.8%	78.1%	9.9%	X	82.1%	80.8%
MI50	71.2%	69.1%	-	X	76.0%	E
IrisPro Gen9	78.6%	X	X	X	80.1%	80.5%

Summary

- SYCL's future is looking bright:
  - Early view of SYCL-2020 shows lots of new HPC-friendly features
    - https://www.iwocl.org/iwocl-2020/conference-program/#panel
  - Support for NVIDIA GPUs added to open-source version of DPC++
  - Critical part of Argonne National Laboratory path to Exascale with Aurora
  - Robust support from/for Arm and AMD the next step
  - Improvements on Intel CPUs needed to help performance

- OpenMP GPU support growing:
  - Improvements to LLVM and GCC
  - Support for Intel GPUs available in Intel oneAPI compiler

- Kokkos continues to provide pragmatic isolation from underlying vendor support decisions:
  - But must wait for Kokkos team or contributors to provide new backends
  - Not open standard so has a high cost of ownership and little shared infrastructure (like LLVM community)

- **What programming model should I use?**
  http://uob-hpc.github.io/2020/05/05/choosing-models.html

- **Performance Portability across Diverse Computer Architectures**
  T. Deakin, S. McIntosh-Smith, J. Price, A. Poenaru, P. Atkinson, C. Popa, J. Salmon, P3HPC at SC 2019.
  https://doi.org/10.1109/P3HPC49587.2019.00006

- **Evaluating the performance of HPC-style SYCL applications**
  T. Deakin, S. McIntosh-Smith, IWOCL 2019.
  https://doi.org/10.1145/3388333.3388643

- **Evaluating attainable memory bandwidth of parallel programming models via BabelStream**
  T. Deakin, J. Price, M. Martineau, S. McIntosh-Smith, IJCSE 2018.
  https://doi.org/10.1504/IJCSE.2017.10011352

Plus others at uob-hpc.github.io/ and hpc.tomdeakin.com and uob-hpc.github.io/SimonMS/

Twitter: @tjdeakin @simonmcs

University of BRISTOL

GW4